You can only do so much with storing information in files. When you need to store large amounts of data, and perform intensive number crunching on that data. In this section, we'll discuss connecting JSP to a MySQL database and perform queries and retrieve data back from the database.

What is JDBC

JDBC stands for Java Database Connectivity, which is a standard Java API for database-independent connectivity between the Java programming language and a wide range of databases. JDBC API is a Java API that can access any kind of tabular data, especially data stored in a Relational Database. The JDBC library includes APIs for each of the tasks mentioned below that are commonly associated with database usage.

- Making a connection to a database.
- Creating SQL or MySQL statements.
- Executing SQL or MySQL queries in the database.
- Viewing & Modifying the resulting records.

After you've installed the appropriate driver, it is time to establish a database connection using JDBC. The programming involved to establish a JDBC connection is fairly simple. Here are these simple five steps −

1. **Import JDBC Packages** − Add import statements to your Java program to import required classes in your Java code. To use the standard JDBC package, which allows you to select, insert, update, and delete data in SQL tables, add the following imports to your source code −

Example

```
import java.sql.* ;  // for standard JDBC programs
import java.math.* ; // for Big Decimal and Big Integer support
```

2. **Register JDBC Driver** − This step causes the JVM to load the desired driver implementation into memory so it can fulfill your JDBC requests. You need to do this registration only once in your program. You can register a driver in ways.

   forName() The most common approach to register a driver is to use Java's **Class. forName()** method, to dynamically load the driver's class file into memory, which automatically registers it. This method is preferable because it allows you to make the driver registration configurable and portable.

Example

```
Class.forName("com.mysql.jdbc.Driver");              // Register for MySQL driver
Class.forName("oracle.jdbc.driver.OracleDriver");    //  Register for oracle driver
```

3. **Database URL Formulation** − This is to create a properly formatted address that points to the database to which you wish to connect. After you've loaded the driver, you can establish a connection using the ***DriverManager.getConnection()*** method. For easy reference

Example

DriverManager.getConnection(*String url, String user, String password*)
Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/aligarh","root","");

4. **Create Connection Object** – Finally, code a call to the DriverManager object's getConnection( ) method to establish actual database connection. The most commonly used form of getConnection() requires you to pass a database URL, a username, and a password

5. **Closing JDBC Connections-** At the end of your JDBC program, it is required explicitly to close all the connections to the database to end each database session. However, if you forget, Java's garbage collector will close the connection when it cleans up stale objects.

   Relying on the garbage collection, especially in database programming, is a very poor programming practice. You should make a habit of always closing the connection with the close() method associated with connection object. To ensure that a connection is closed, you could provide a 'finally' block in your code. A finally block always executes, regardless of an exception occurs or not.

   To close the above opened connection, you should call ***close()*** method as follows

Example

```
con.close();
```

Connection for MYSQL Database management tool.

The first thing that we need to do before we can interact with any database, is to open up a connection to that database server. The database handler is then used to select the active database to use.

To connect Java application with the MySQL database, we need to follow 5 following steps.

1. To connect java application with the MySQL database, mysqlconnector.jar file is required to be loaded. Paste the mysqlconnector.jar file in ProjectName/WEB-INF/lib/
2. Driver class: The driver class for the MySQL database is com.mysql.jdbc.Driver.
3. Connection URL: The connection URL for the MySQL database is jdbc:mysql://localhost:3306/DatabaseName1 where JDBC is the API, MySQL is the database, localhost is the server name on which MySQL is running, we may also use IP address, 3306 is the port number and DatabaseName1 is the database name. We may use any database, in such case, we need to replace the DatabaseName1 with our database name.
4. Username: The default username for the MySQL database is root.
5. Password: It is the password given by the user at the time of installing the MySQL database. In this example, we are going to use root as the password.

Copy Right
http://www.sirmasood.com

Database Connection with MySQL Tool in JSP

In this example we are using MySQL as the database. So we need to know following information's for the MySQL database: Let's first create a database that name is **Aligarh** in MySQL database management tool.

```
<%@page import = "java.sql.*,util.*" %>
<html>
<body>
<h1> Database Connect in JSP </h1>
<%
 try
 {
    Class.forName("com.mysql.jdbc.Driver");
    Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/aligarh","root","");

      out.println("MySQL Database Sucessfully connected...");
      con.close();
 }
 catch(Exception e)
 {
          out.println("MySQL Database Not Connected Error: "+e);
 }
%>
</body>
</html>
```

What is CRUD OPERATION?

Within computer programming, the acronym CRUD stands for create, read, update and delete. Most applications have some form of CRUD functionality. In fact, every programmer has to deal with CRUD at some point. A CRUD application is one that utilizes forms to retrieve and return data from a database

Example of add new record into database table.

The INSERT INTO statement is used to add new records to a MySQL table: before we created an empty table named "Students" with three columns: "Roll", "Name" and "Father Name". Now, let us fill the table with data.

Copy Right
http://www.sirmasood.com

A Prepared Statement is a pre-compiled SQL statement. It is a sub interface of Statement. Prepared Statement objects have some useful additional features than Statement objects. Instead of hard coding queries, PreparedStatement object provides a feature to execute a parameterized query.

```
<%@page import = "java.sql.*,util.*" %>
<html>
<body>
<h1> Show And Save Information </h1>
<%
  int Roll = 103;
  String Name = "Muhammad Ali";
  String Fname = "Muhammad Ahmed";
 try
 {
   Class.forName("com.mysql.jdbc.Driver");
    Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/aligarh","root","");
    PreparedStatement ps = con.prepareStatement("insert into students(Roll,Name,FatherName) values ( ?,?,? ); ");
    ps.setInt(1,Roll);
    ps.setString(2,Name);
    ps.setString(3,Fname);

    int x = ps.executeUpdate();
            out.println("Record Has Been Saved.. "+x);

 }
 catch(Exception e)
 {
            out.println("Error: "+e);
 }
%>
</body></html>
```

If Add New Record Has Been Saved then display below screen

If Add New Record Has not Been saved or any error then display below screen

The above example will insert new Record in the Student table. Here we use A Prepared Statement is a pre-compiled SQL statement. It is a sub interface of Statement. Prepared Statement objects have some useful

additional features than Statement objects. Instead of hard coding queries, PreparedStatement object provides a feature to execute a parameterized query.

PreparedStatement ps = con.prepareStatement ("insert into students(Roll,Name,FatherName) values (?,?,?); ");

As you can see, we are passing parameter (?) for the values. Its value will be set by calling the setter methods of PreparedStatement.

Methods of PreparedStatement interface

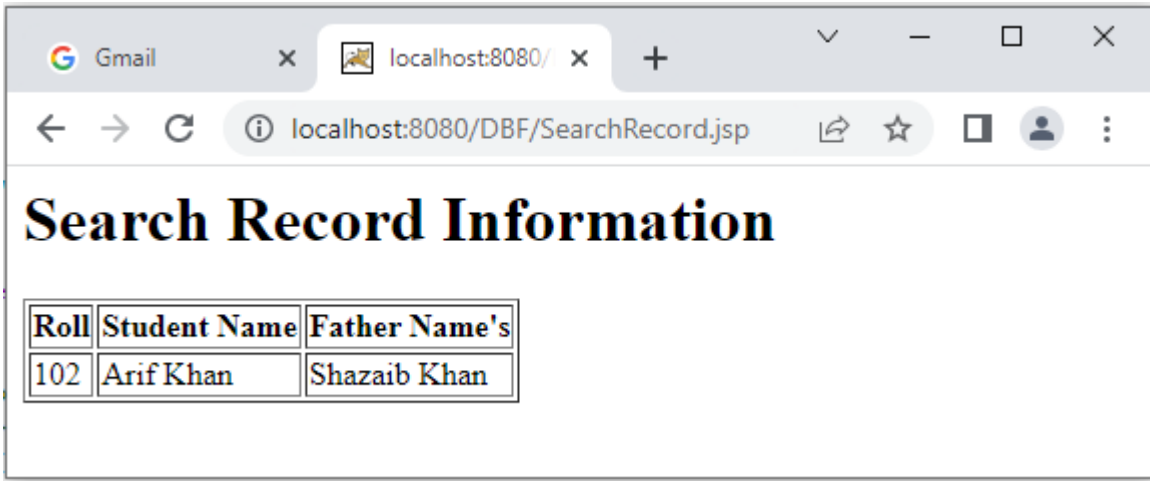
The important methods of ***PreparedStatement()*** interface are given below:

Method	Description
public void setInt(int paramIndex, int value)	sets the integer value to the given parameter index.
public void setString(int paramIndex, String value)	sets the String value to the given parameter index.
public void setFloat(int paramIndex, float value)	sets the float value to the given parameter index.
public void setDouble(int paramIndex, double value)	sets the double value to the given parameter index.
public int executeUpdate()	executes the query. It is used for create, drop, insert, update, delete etc.
public ResultSet executeQuery()	executes the select query. It returns an instance of ResultSet.

Example of Display Single/Search Record using by JSP.

```
<%@page import = "java.sql.*,util.*" %>
<html>
<body>
<h1> Search Record Information </h1>
<%
  int Roll = 102;

 try
 {
        Class.forName("com.mysql.jdbc.Driver");
   Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/aligarh","root","");

   PreparedStatement pst = con.prepareStatement("SELECT * FROM  students WHERE Roll = ? ");
        pst.setInt(1,Roll);
   ResultSet rs = pst.executeQuery();
        out.print("<Table border=1");
   out.print("<TR><th>Roll </th><th>Student Name</th> <th> Father Name's</th></tr> ");
        if (rs.next())
        {
```

```
    out.print("<TR>");
    out.print("<TD>" + rs.getInt("Roll") + "</TD>");
    out.print("<TD>" + rs.getString("Name") + "</TD>");
    out.print("<TD>" + rs.getString("FatherName") + "</TD>");
    out.print("</TR>");
    out.print("</Table>");
          }
          else
            out.print(" <h2 style='color:red'> Record is not found </h2>");
 }
 catch(Exception e)
 {
          out.println("<h3 style=çolor:red'> Error: "+e+"<h3>");
 }

%>
</body>
</html>
```

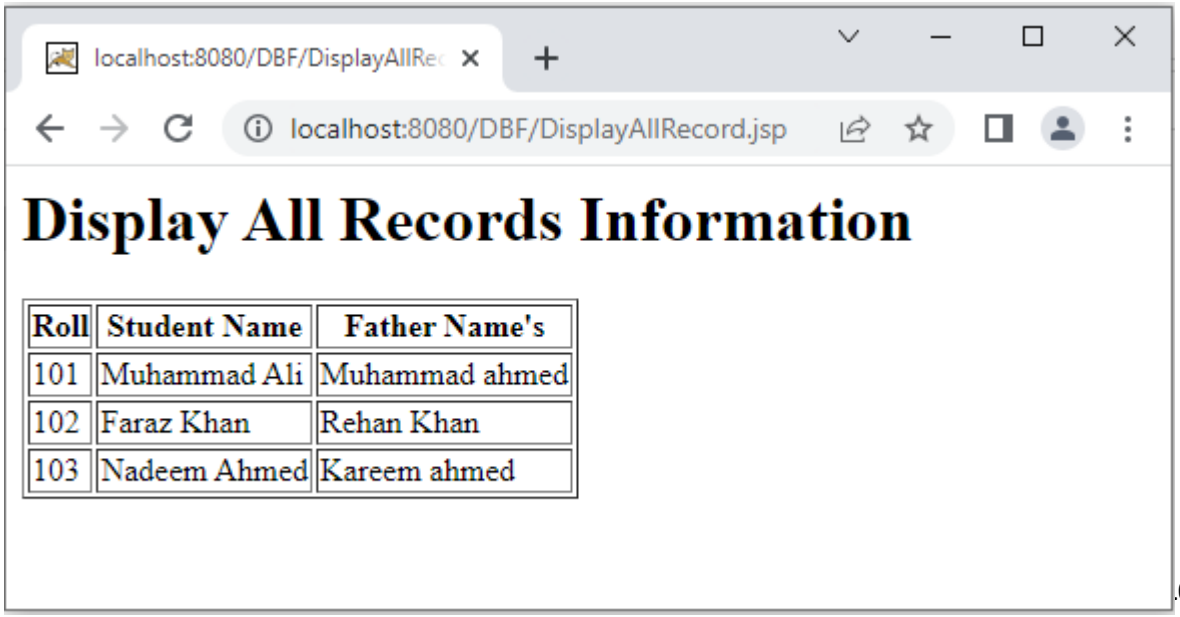


Select/Retrieve Data from database of MySQL using by JSP.
Example of Display All Records

First, we set up an SQL query that selects the Roll, Name and FatherName columns from the Students table. The next line of code runs the query and puts the resulting data into a variable called **rs**. Then, the from ResultSet class.
If there are more than zero rows returned, the ResultSet class puts all the results into an array that we can loop through. The while() loop, loops through the result set and outputs the data from the Roll, Name and FatherName columns.

```
<%@page import = "java.sql.*,util.*" %>
<html>
<body>
<h1> Display All Records Information </h1>
```

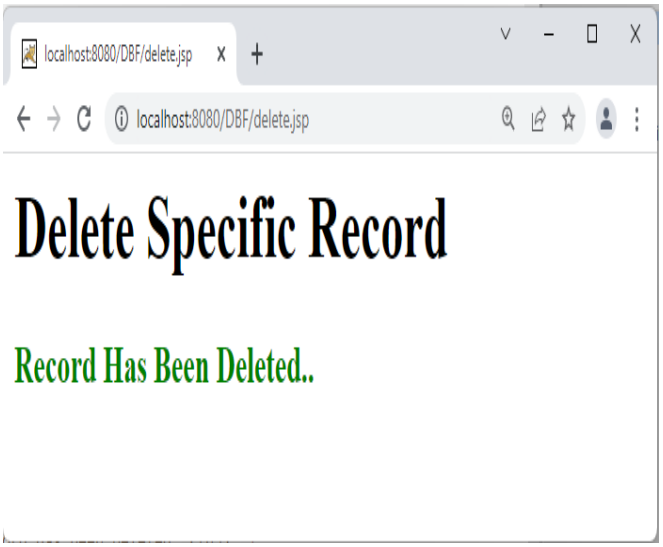
```
<%

 try
 {
   Class.forName("com.mysql.jdbc.Driver");
    Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/aligarh","root","");

    PreparedStatement pst = con.prepareStatement("SELECT * FROM  students  ");

    ResultSet rs = pst.executeQuery();
    out.print("<Table border=1");
    out.print("<TR><th>Roll </th><th>Student Name</th> <th> Father Name's</th></TR> ");
          while (rs.next())
          {

     out.print("<TR>");
     out.print("<TD>" + rs.getInt("Roll") + "</TD>");
     out.print("<TD>" + rs.getString("Name") + "</TD>");
     out.print("<TD>" + rs.getString("FatherName") + "</TD>");
     out.print("</TR>");

          }
          out.print("</Table>");

 }
 catch(Exception e)
 {
          out.println("<h3 style=çolor:red'> Error: "+e+"<h3>");
 }

%>
</body>
</html>
```

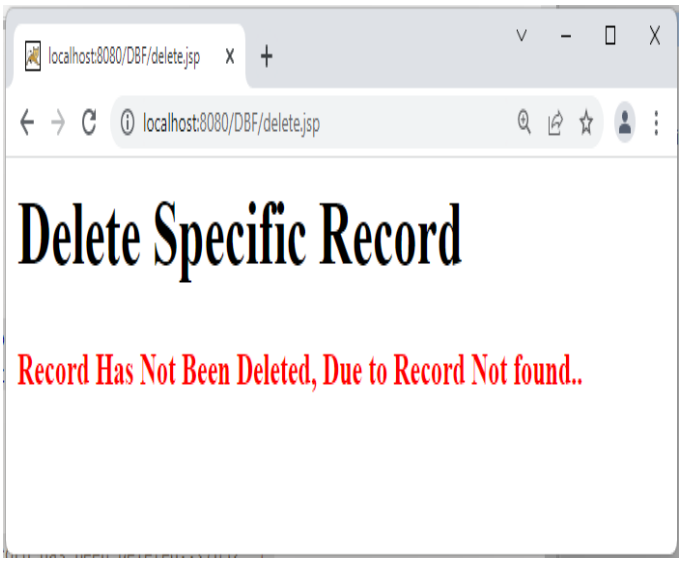


Example of Delete Record using by JSP.

```
<%@page import = "java.sql.*,util.*" %>

<html>
<body>
<h1> Delete Specific Record </h1>
<%

  int Roll = 101;

    try
  {
      Class.forName("com.mysql.jdbc.Driver");
      Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/aligarh","root","");
      PreparedStatement ps = con.prepareStatement("delete from students where  Roll = ?");
      ps.setInt(1,Roll);

      int x = ps.executeUpdate();
     if (x==1)
         out.println("<h3 style='color:green'>Record Has Been Deleted..</h3> ");
    else
        out.println("<h4 style='color:red'>Record Has Not Been Deleted, Due to Record Not found.. </h4>");

 }
 catch(Exception e)
 {
            out.println("Error: "+e);
 }

%>
</body>
</html>
```

The above example will Delete Record of Roll 101 in the Students table. If record is exist then display Bellow left Screen otherwise display below right screen





Copy Right
http://www.sirmasood.com

Example of Update Record using by JSP.

```
<%@page import = "java.sql.*,util.*" %>
<html>
<body>
<h1> Update Specific Record </h1>
<%


 int Roll = 101;
 String Name=" Fatima";
 String Fname="Muhammad";

 try
 {
  Class.forName("com.mysql.jdbc.Driver");
  Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/aligarh","root","");
  PreparedStatement ps = con.prepareStatement("update students set Name = ?  , FatherName = ? where  Roll = ?  ");
    ps.setInt(1,Roll);
    ps.setString(2,Name);
    ps.setString(3,Fname);
    int x = ps.executeUpdate();
     if (x==1)
         out.println("<h3 style='color:green'>Record Has Been Updated..</h3> ");
    else
        out.println("<h4 style='color:red'>Record Has Not Been Updated, Due to Record Not found.. </h4>");

 }
 catch(Exception e)
 {
              out.println("Error: "+e);
 }
%>
</body>
</html>
```

The above example will replace Name and FatherName Record of Roll 101 in the Student table. If record is exist then display Bellows Screen

Exercise

Theory Questions.

1) What do you mean by CRUD operation?
2) Write step Connection for MYSQL Database in JSP with detail.

Practical Questions.

1) Write code of JSP for use connect to the database name "***Education***".
2) You write a program to input ***Roll, Name*** and ***Father Name*** for add new record in the table "***Students***" of database "***Education***".
3) Write a JSP program to input **Roll** number and delete record according to input roll number from table ***"Students"*** of database "***Education***".
4) Write a JSP program to display student record according to specific input ***Roll*** number from table "***Students***" of database "***Education***".
5) Write a JSP program to Display All records from table ***Students*** of database name is "***Education"***. Display all records are following form.

S.No	Roll	Name	Father Name
1	11	Muhammad Ali	Imran Khan
2	120	Farooq Ahmed	Usman Ahmed
3	7	Asif Ali	Rehan Ali

Objective and MCQ's

1) Which method sets the integer value to the given parameter index..
   a) setInt()
   b) setString()
   c) setFloat()
   d) setGet().
2) Which method sets the String value to the given parameter index.
   a) setInt()
   b) setString()
   c) setFloat()
   d) setGet()
3) Executes the select query. It returns an instance of ResultSet.
   a) executeQuery()
   b) executeUpdate()
   c) executeRun()
   d) Nothing

Copy Right
http://www.sirmasood.com

4) As you can see, we are passing parameter ( ) for the values. Its value will be set by calling the setter methods of PreparedStatement.
   a) &
   b) #)
   c) !
   d) ?

5) _____ Object provides a feature to execute a parameterized query.
   a) PreparedStatemen(t)
   b) PreparedCommand()
   c) ExecuteQuery()
   d) StatementPrepared()

6) which MySQL is running, we may also use IP address, _____ is the port number and is the database name
   a) 8080
   b) 3301
   c) 8083
   d) 3306

Copy Right
http://www.sirmasood.com